1 A style guide for Python

Every language, programming and otherwise, develops its own style designed to enhance comprehension. Python is no different. And particularly in the case of Python, it is designed to be read, often multiple times.

The Python Software Foundation (PSF) has produced a style guide for writing code and is the first place to go when seeking definitive online documentation. This can be found at:
https://www.python.org/dev/peps/pep-0008/#introduction

The documentation is at https://docs.python.org/3/ but there are many other places to find help, too.

Selections from the Style Guide:

- Use 4 spaces for every indentation level.
- Spaces are the preferred indentation method.
- Python 3 disallows mixing the use of tabs and spaces for indentation.
- Limit all lines to a maximum of 79 characters.
- For flowing long blocks of text, the line length should be limited to 72 characters.
- Avoid trailing whitespace anywhere.
- Never use the characters ‘l’ (lowercase letter el), ‘O’ (uppercase letter oh), or ‘I’ (uppercase letter eye) as single character variable names.

Task 0: Go to the Python section of our course web-page and familiarise yourself with the abbreviated style guide:
http://eg.bucknell.edu/~koutsits/Ph222/

Task 1: Review your submitted program from last week (Lab 1), and modify it (if necessary) so that it is consistent with the Python style guide.

2 Conditionals

Your program can do things only if certain conditions are met. Try this small program. Any indented lines after the if ...: will be executed if the condition is met.

```python
my_number = input("Enter a whole number no greater than ten: ")
my_number = int(my_number)
if my_number > 10:
    print("You entered a number greater than ten."
else:
    print("You entered a valid number: " + str(my_number))
```

Do you understand what is going on here?

**Note: The indentation is essential.** Other boolean operators include: == , >= , and != . Find out what they do and write it down.
Task 2: Modify the above program so that it asks you to enter any integer and tells you whether you entered a positive number, a negative number, or zero. (Hint: look up the use of elif in an if statement.)

Demonstrate your program to your instructor.

3 Loops

The first line in a loop is a conditional statement. Everything on the following indented lines will be executed repeatedly as long as the condition is true. The IDLE environment will automatically indent the first line after a conditional statement, but you must “unindent” when you get to the end of the statements that are to be repeated.

Example 1:

```python
i = 1
while i<11:
    print(i)
    i += 1
```

Example 2:

```python
i = 1
while True:
    print(i)
    i += 1
```

In Example 2, the argument to the while loop is simply the constant value True, so the loop is continually executed. You can break the infinite loop in the last example with a CTRL-C (simulta-neously pressing the CTRL and the C keys).

Instead of a while loop, the above code can be implemented through a for loop as well.

Example 3:

```python
for i in range(11):
    print(i)
```

Task 3: Return to your program that calculates the height of a dropped ball. Modify your script so that it prints the height of the ball at regular time intervals of \( t = 0, 1, 2, 3, \ldots \) seconds until the ball reaches the ground. Be sure to drop your ball from a height greater than 100 m.

4 Functions

Python treats functions just like any other variable. This means that you can store functions in other variables or sequences, and even pass those functions to other functions. For example, if we need to frequently evaluate \( f(x) = x^4 + 5x^2 - 3x \), it would be much easier to define it as a function:

```python
def myfunction(xvalue):
    f_value = xvalue**4 + 5*xvalue**2 - 3*xvalue
    return f_value
```
When we need $f(x)$ later in the program, all we need to do is call on `myfunction`.

```python
for i in range(3)
    print myfunction(i)
```

Your function may have multiple inputs. Suppose you want a kinetic energy function:

```python
def kinetic_E(m, v):
    the_answer = 0.5 * m * v**2
    return the_answer
```

**Task 4:** Type the following program into a file and run it.

```python
N = 10000
a = 1

def f(x,k):
    return 0.5*(x + k/x)

for j in range(10):
    a = f(a,N)
    print(a)
```

Based on the output, and what you have learned in calculus, you may guess that the program uses Newton’s method to compute the square root of $N$. Try it with another value of $N$.

Instead of looping 10 times, or some other fixed number of times, it would be better to continue until the value stops changing only an acceptably small amount.

Modify the program so that it uses a `while` loop, and have it continue looping until the difference between the previous and the current values of ‘$a$’ differ by less than 0.0001. You will find it helpful to introduce new variables such as `$a_{old}$` and `$a_{new}$`, which are the values of `$a$` from the previous and the current iteration of your loop. The top of your loop might look something like:

```python
while a_new - a_old < 0.0001:
```

You will find it helpful to include a `print` statement inside the loop to help you keep track of what the program is doing.

Demonstrate your program to your instructor.

## 5 Assignment: The Guessing Game

Go to our course website: http://eg.bucknell.edu/~koutslts/Ph222/ and download the file called `guessinggame.txt`. The instructions for this exercise are contained in the file.

NB: Rename it `guessinggame.py` on your computer before you begin.

**Final report:**

Submit your program from the Guessing Game assignment. Place it in the ‘koutslts’ drop_box before 4:30 p.m. on Monday. Be sure to identify your code appropriately.